Anisotropic Simplicial Meshing Using Local Convex Functions

Xiao-Ming Fu[*†] Yang Liu[†] John Snyder[†] Baining Guo[†*]
[*]University of Science and Technology of China [†]Microsoft Research

Figure 1: *Anisotropic meshes generated by our method. Left: 2D meshing. The anisotropic metric is defined as the Hessian of an analytic function evincing a large range of anisotropy ratios ($\lambda \in [1.9, 394.4]$). Zoom in on the image to see meshing details. Middle: 3D surface meshing of the Happy Buddha from curvature tensors estimated from a high-resolution reference mesh (115474 vertices). Our relaxation produces a high quality result (63284 vertices) starting with an initial low-resolution mesh (5000 vertices). Right: volumetric meshing in a 3D cube. Anisotropy changes substantially ($\lambda \in [1, 40]$) and rapidly over the domain. The lower image shows a cross-section.*

Abstract

We present a novel method to generate high-quality simplicial meshes with specified anisotropy. Given a surface or volumetric domain equipped with a Riemannian metric that encodes the desired anisotropy, we transform the problem to one of functional approximation. We construct a convex function over each mesh simplex whose Hessian locally matches the Riemannian metric, and iteratively adapt vertex positions and mesh connectivity to minimize the difference between the target convex functions and their piecewise-linear interpolation over the mesh. Our method generalizes optimal Delaunay triangulation and leads to a simple and efficient algorithm. We demonstrate its quality and speed compared to state-of-the-art methods on a variety of domains and metrics.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Geometric algorithms, languages, and systems

Keywords: anisotropic meshing, locally convex triangulation

Links: DL PDF

1 Introduction

The term "anisotropy" characterizes the spatially-varying sizes, orientations and aspect ratios of mesh elements. Controlling anisotropy when generating meshes is essential in a wide variety of applications including geometric modeling, physical simulation, field visualization, and mechanical engineering.

Many anisotropic meshing techniques have been proposed in the last two decades. Extending the circumsphere property generalizes Delaunay triangulation to handle anisotropic meshing [Thompson et al. 1998; Frey and George 2008]. Particle-based approaches [Shimada et al. 2000; Zhong et al. 2013] apply repulsive forces that conform to the desired anisotropic distance between vertices. The Voronoi diagram can also be generalized in terms of anisotropic geodesic distance, yielding the anisotropic Voronoi diagram (AVD). Refining [Labelle and Shewchuk 2003] or optimizing [Du and Wang 2005] the AVD and taking its dual yields yet another strategy.

These methods directly sample or adjust vertices so that, under the inverse transformation induced by the Riemannian metric, the mesh has nearly unit-length edges and equilateral simplices. They have two main limitations. First, they are expensive. It is especially costly to construct and refine the AVD, accumulate forces from many neighboring particles, or evaluate Riemannian geodesic distance in complex domains. Second, their output leaves room for improvement in reproducing the targeted anisotropy. A few methods tackle this problem by bounding specific quality metrics like a radius-edge or sliver ratio [Labelle and Shewchuk 2003; Boissonnat et al. 2008b; Boissonnat et al. 2014], but only work for smooth anisotropic fields in domains without sharp boundary features. Despite these guarantees, overall mesh quality remains insufficient, both visually and in terms of simple objective measures of its match to the specified anisotropy (see comparisons in Section 4).

A different approach is to numerically solve a PDE so that the solution's Hessian matches the desired anisotropy. An optimal anisotropic mesh can be obtained by minimizing the difference between the exact solution and its discrete interpolation over the mesh [Chen et al. 2007]. In fact, the optimal result specializes to a *regular triangulation* when its anisotropy is specified as the Hessian of a global convex function and L^p difference is minimized [Chen and Xu 2004]. These ideas have been applied to 2D/3D isotropic meshing in which the regular triangulation further degenerates to a Delaunay triangulation [Alliez et al. 2005; Chen and Holst 2011]. However, such a convex function does not in general exist for anisotropic metrics [Boissonnat et al. 2008a].

We extend the PDE-based approach to introduce a new method for anisotropic simplicial meshing of surfaces and volumes. The key idea is to construct convex functions that *locally* match the specified Riemannian metric, defined on each simplex. We then relax a mesh to minimize the integrated difference between each function and its linearly-interpolated discretization, summed over all simplices. Our contributions are summarized as follows.

- We propose *locally convex triangulation* (LCT), a new generalization of optimal Delaunay triangulation to anisotropic metrics.
- We present a simple algorithm that iteratively optimizes and refines the mesh to satisfy the anisotropy target and bound geometric error. Our method works for highly-varying anisotropy and sharp boundary features, improving both the output quality and computational efficiency of anisotropic meshing.

2 Related Work

Anisotropic meshing defines a Riemannian tensor field $\mathcal{M}$ over a given domain $\Omega \subset \mathbb{R}^d$. The anisotropy associated with a point $\mathbf{p} \in \Omega$ is represented as a $d \times d$ positive-definite matrix $\mathbf{M}(\mathbf{p})$, called the *augmented metric*. Its eigen-decomposition $\mathbf{Q}\,\mathbf{\Lambda}\,\mathbf{Q}^T$ characterizes the desired stretch directions and magnitudes of mesh elements, in terms of a rotation matrix $\mathbf{Q}$ and diagonal matrix $\mathbf{\Lambda}$. The *anisotropy ratio* is defined as $\lambda \triangleq \sqrt{\max \lambda_i / \min \lambda_i}$ where λ_i are the eigenvalues of $\mathbf{M}$ (diagonal elements of $\mathbf{\Lambda}$). The length of the segment between two points $\mathbf{p}, \mathbf{q} \in \Omega$ with respect to the augmented metric is defined as $\int_0^1 \sqrt{(\mathbf{p}-\mathbf{q})^T\,\mathbf{M}\,(t\mathbf{p}+(1-t)\mathbf{q})\;(\mathbf{p}-\mathbf{q})}\,\mathrm{d}t$ and approximated by $\sqrt{(\mathbf{p}-\mathbf{q})^T\,(\mathbf{M_p}+\mathbf{M_q})/2\,(\mathbf{p}-\mathbf{q})}$. An anisotropic simplex can be transformed to a regular simplex in Euclidean space via $\mathbf{\Lambda}^{-1/2}\,\mathbf{Q}^T$, called the *inverse transformation*.

Delaunay-based methods Isotropic meshes can be generated by repeatedly inserting Steiner points in poor-quality elements and recomputing the Delaunay triangulation. This method can be extended to anisotropic meshing by modifying the definition of Steiner points according to the anisotropic Delaunay kernel [Frey and George 2008; Dobrzynski and Frey 2008]. Other methods refine the triangulation via local operations based on mesh quality [Hecht 1998; Jiao et al. 2010]. Boissonnat et al. propose a variant called the *locally uniform anisotropic mesh*, developing a vertex insertion algorithm that guarantees certain measures of mesh quality [2008b; 2011; 2014] when the tensor field and domain boundary are both smooth.

Anisotropic Voronoi methods A set of points $\{\mathbf{p}_i \in \mathbb{R}^d\}_{i=1}^n$ partitions Euclidean space $\mathbb{R}^d$ into a set of disjoint polyhedral cells $\{\Omega_i\}_{i=1}^n$ where $\Omega_i = \{\mathbf{x} \in \mathbb{R}^d \mid D(\mathbf{x},\mathbf{p}_i) \le D(\mathbf{x},\mathbf{p}_j)\,\forall\, j \ne i\}$ and D is the distance function. If $D(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|^2$, the partition reverts to an ordinary Voronoi diagram. When it instead measures Riemannian distance on $(\Omega, \mathcal{M})$, denoted $D_{\mathcal{M}}$, the partition becomes an *anisotropic Voronoi diagram* (AVD) whose dual, called the *anisotropic Delaunay triangulation* (ADT), is a simplicial mesh under certain conditions. By refining an initial AVD through vertex insertion [Labelle and Shewchuk 2003; Cheng et al. 2006], or vertex optimization via an anisotropic centroidal Voronoi tessellation (ACVT) energy function, $\sum_i \int_{\Omega_i} \|D_{\mathcal{M}}(\mathbf{x},\mathbf{p}_i)\|^2\,\mathrm{d}\mathbf{x}$ or its variants [Du and Wang 2005; Valette et al. 2008; Lévy and Liu 2010; Lévy and Bonneel 2012], the ADT generated attempts to match the desired anisotropy. However, refinement only works in 2D, while ACVT methods are impractically expensive as they construct the AVD many times. The ACVT dual is also not guaranteed to be manifold [Canas and Gortler 2011].

Particle-based methods Repositioning points by repulsive or spring forces [Shimada et al. 2000; Persson and Strang 2004] is a popular technique in mesh generation. Force can be related to a Riemannian metric by modeling energy between two edge points $\mathbf{p}_i$ and $\mathbf{p}_j$, via e.g. $\exp\left(-\frac{D^2_{\mathcal{M}}(\mathbf{p}_i,\mathbf{p}_j)}{4\sigma^2}\right)$ as in [Zhong et al. 2013]. Minimizing this energy sum over edges and projecting points back onto the domain redistributes the points until forces reach equilibrium. A simplicial mesh can then be generated by anisotropic Delaunay triangulation [Frey and George 2008] or by extraction from the AVD. The main advantage of particle-based methods is their simple, meshless flexibility. However, the choice of kernel width σ and other parameters greatly affects output quality especially when the anisotropy varies widely. As with anisotropic Voronoi methods, postprocessing is necessary to obtain a manifold ADT output.

Functional methods Anisotropic mesh quality can be measured by the error between a PDE solution u and its piecewise linear nodal interpolation on the mesh $\hat{u}$ [Zienkiewicz et al. 2005]. Chen et al. propose *optimal Delaunay triangulation* (ODT) minimizing the error function $\|u-\hat{u}\|_{L^p(\Omega)}$ [2004; 2007]. ODT has been successfully applied to isotropic simplicial meshing [Alliez et al. 2005; Tournois et al. 2009b; Chen and Holst 2011] where $p=1$ and $u(\mathbf{x}) = \mathbf{x}^2$. When the anisotropy is the Hessian of a convex function u, ODT naturally extends to anisotropic meshing and the triangulation becomes regular (also called the weighted-Delaunay triangulation). This property has been exploited in other applications including self-supporting surface design [Liu et al. 2013], electric impedance tomography [Desbrun et al. 2013] and other geometric processing applications [Mullen et al. 2011; Goes et al. 2014]. However, a Riemannian metric is not in general the Hessian of any global function [Boissonnat et al. 2008a; Amari and Armstrong 2014]. Chen [2004] proposes a method to apply ODT *locally* on each one-ring neighborhood of a vertex but his method fails as discussed in Section 4.1. Our approach overcomes these problems.

3 Locally Convex Triangulation

We review optimal Delaunay triangulation in Section 3.1, then introduce our locally convex triangulation (LCT) energy and study its relationship with anisotropic mesh quality in 3.2. We develop an efficient algorithm for minimizing LCT energy in 3.3, and combine it with other criteria in 3.4.

3.1 Optimal Delaunay triangulation

Chen et al. proposed optimal Delaunay triangulation (ODT) which considers anisotropic meshing as a functional approximation problem [2004; 2007]. Denote a simplicial triangulation as $\mathcal{T} = \bigcup_{k=1}^N \tau_k$ where τ_k is the k-th simplex in $\mathcal{T}$. Given a function $u : \mathbb{R}^d \to \mathbb{R}$ defined over a domain $\Omega \in \mathbb{R}^d$, $d \ge 1$, ODT finds a triangulation $\mathcal{T}$ on Ω with a given number of vertices that minimizes L^p error between u and its piecewise linear interpolation over vertices of $\mathcal{T}$, $\hat{u}$. More precisely, ODT error is defined as

$$E_{ODT,u,p} \triangleq \|\hat{u}-u\|_{L^p} = \left(\sum_{\tau\in\mathcal{T}} \int_\tau |\hat{u}(\mathbf{x})-u(\mathbf{x})|^p\,\mathrm{d}\mathbf{x}\right)^{1/p}.$$

When u is a convex function, $E_{ODT,u,1}$ measures the volume difference between u and $\hat{u}$, and its minimizer $\hat{u}$ yields a tight and strictly upper envelope.

Denote the Hessian of u as $\partial^2 u$, with eigen-decomposition $\partial^2 u = Q\,\Lambda\,Q^T$. The function u induces the Riemannian metric $\mathbf{M} = (\det M)^{-\frac{1}{2p+d}}\,M$ in Ω where, to ensure positive-definiteness, M is computed as the *majorant* of $\partial^2 u$:

$$M \triangleq Q\,|\Lambda|\,Q^T + \delta I \tag{1}$$

for a small value $\delta \geq 0$.

Minimizing $E_{ODT,\|\mathbf{x}\|^2,1}$ leads to an isotropic mesh $\mathcal{T}$ which is also Delaunay. The minimization executes two steps iteratively [Alliez et al. 2005]. First, it updates mesh vertices via

$$\mathbf{v}_i^\star := \frac{1}{|\Omega_i|}\sum_{\tau\in\Omega_i} |\tau|\,\mathbf{c}_\tau, \tag{2}$$

where Ω_i represents the set of simplices incident on vertex $\mathbf{v}_i$, $|\cdot|$ is the area/volume of a simplex or set of simplices, and $\mathbf{c}_\tau$ is the circumcenter of τ. Second, it rebuilds mesh connectivity via constrained Delaunay triangulation.

For a general convex function u, one can obtain the anisotropic mesh by optimizing $E_{ODT,u,1}$ as shown in [Chen and Xu 2004; Liu et al. 2013; Desbrun et al. 2013]. The optimal triangulation leads to a constrained regular triangulation whose power weight function is given by $w(\mathbf{x}) = \mathbf{x}^2 - 2u(\mathbf{x})$. The optimal position of vertex $\mathbf{v}_i$ satisfies $\mathbf{v}_i - \frac{\nabla w(\mathbf{v}_i)}{2} = \frac{1}{|\Omega_i|}\sum_{\tau\in\Omega_i}|\tau|\,\mathbf{c}_\tau^\star$ where $\mathbf{c}_\tau^\star$ is the weighted circumcenter of simplex τ in the regular triangulation. Eq. 2 represents the degenerate case in which $w(\mathbf{x})$ is a constant.

3.2 LCT energy

ODT provides a simple solution for anisotropic meshing as long as the metric is the Hessian of a convex function. This is not the case for general Riemannian tensor fields. We observe that the anisotropy inside a single simplex can be approximated locally by its own convex function. An ODT-like energy can then be constructed to measure the interpolation error locally. This simple idea lets us generalize ODT to deal with general Riemannian metrics.

Given a domain Ω equipped with a Riemannian tensor field $\mathcal{M}$ and a simplicial mesh $\mathcal{T}$ discretizing Ω, we define a convex quadratic function u_τ on each simplex $\tau \in \mathcal{T}$. Its Hessian matches the average Hessian of $\mathcal{M}$ over the simplex; i.e., $\partial^2 u_\tau = H_\tau \triangleq \frac{\int_\tau \mathbf{M}(\mathbf{x})\,\mathrm{d}\mathbf{x}}{|\tau|}$. The *simplex metric tensor*, H_τ, is computed using Gaussian quadrature. We use the d-point quadrature rule (simple average over simplex vertices) in our implementation. For extreme anisotropy, adaptive numerical integration [Genz and Cools 2003] would likely reduce potential undersampling. The local convex function is given by

$$u_\tau(\mathbf{x}) = \frac{1}{2}\,\mathbf{x}^T\,H_\tau\,\mathbf{x}. \tag{3}$$

As with ODT, we then measure the error between u_τ and its piecewise linear interpolation over vertices of the simplex τ, denoted $\hat{u}_\tau$, yielding the error function

$$E_{\tau,p} \triangleq \int_\tau |\hat{u}_\tau - u_\tau|^p\,\mathrm{d}\mathbf{x}. \tag{4}$$

Summing $E_{\tau,p}$ over all $\mathcal{T}$, we obtain the locally convex triangulation (LCT) energy function: $E_{LCT,p}(\mathcal{T}) \triangleq \left(\sum_{\tau\in\mathcal{T}} E_{\tau,p}\right)^{1/p}$. We call $\mathcal{T}$ an LCT mesh if $E_{LCT,p}(\mathcal{T})$ reaches a local minimum. We restrict $p = 1$, hereafter dropping the p subscript from the energy.

Assuming u_τ and $\hat{u}_\tau$ are convex (i.e., H_τ is positive-definite), $\hat{u}_\tau$ is a strictly upper envelope for u_τ and the absolute values vanish in Eq. 4. We can substitute u_τ from Eq. 3 into Eq. 4 to obtain

$$E_\tau = \frac{|\tau|\,\sum_{j<k}(\mathbf{p}_j - \mathbf{p}_k)^T\,H_\tau\,(\mathbf{p}_j - \mathbf{p}_k)}{2(d+1)(d+2)}, \tag{5}$$

for a simplex τ with vertices $\mathbf{p}_0, \ldots, \mathbf{p}_d$.

Remark: Chen [2004] defined a mesh quality metric $Q(\mathcal{T}, G, 1) = \frac{1}{d+1}\sum_{i=1}^N \int_{\Omega_i}(\mathbf{p}-\mathbf{p}_i)^T\,G\,(\mathbf{p}-\mathbf{p}_i)\,\mathrm{d}\mathbf{p}$ for a simplicial mesh $\mathcal{T}$ with Riemannian metric G, where N is the number of mesh vertices. Our E_{LCT} is consistent with their definition when $G|_\tau = H_\tau$, as can be seen by verifying that $2(d+3)\,E_{LCT}(\mathcal{T}) = (d+1)\,Q(\mathcal{T}, G, 1)$.

3.3 LCT optimization

Optimizing LCT energy is analogous to ODT. Given an initial mesh, it takes three steps: (1) compute H_τ on each simplex; (2) update vertices by their locally optimal locations; (3) update mesh connectivity. These three steps execute iteratively until the change in energy gets small enough or the maximal iteration number is reached.

Vertex update We use Newton's method and assume the metric tensor $\mathbf{M}$ changes slowly over the neighborhood of a mesh vertex so that its spatial derivative can be ignored. Given the (fixed) simplex metric tensor H_τ, energy and its first and second derivatives with respect to a vertex $\mathbf{p}_k \in \tau$ can be calculated via:

$$E_\tau = \frac{|\tau|\,F(\tau)}{(d+1)(d+2)},$$

$$\partial_{\mathbf{p}_k} E_\tau = \frac{1}{(d+1)(d+2)}\,\left(|\tau|\,\mathbf{s}_k + F(\tau)\,\mathbf{n}_k\right),$$

$$\mathbf{h}_{\tau,\mathbf{p}_k} \triangleq \partial^2_{\mathbf{p}_k} E_\tau = \frac{1}{(d+1)(d+2)}(|\tau|\,H_\tau + \mathbf{s}_k\,\mathbf{n}_k^T + \mathbf{n}_k\,\mathbf{s}_k^T),$$

where $F(\tau) = 1/2\,\sum_{j<k}(\mathbf{p}_k - \mathbf{p}_j)^T\,H_\tau\,(\mathbf{p}_k - \mathbf{p}_j)$, $\mathbf{s}_k = H_\tau\,\sum_{j\neq k}(\mathbf{p}_k - \mathbf{p}_j)$, and $\mathbf{n}_k = \partial_{\mathbf{p}_k}|\tau|$. Note that $\mathbf{h}_{\tau,\mathbf{p}_k}$ is not always positive-definite. When applying Newton's method, we drop the last two terms to obtain $\mathbf{h}_{\tau,\mathbf{p}_k} \approx |\tau|\,H_\tau/((d+2)(d+1))$, ensuring a positive-definite result. This strategy works well in practice. In particular, if the simplex metric tensors of all its one-ring simplices $\Omega_\mathbf{p}$ are identical at an interior vertex $\mathbf{p}$, then the last two terms above vanish in the sum of second derivatives over $\Omega_\mathbf{p}$.

Our update rule replaces each mesh vertex $\mathbf{p}$ with a new location $\mathbf{p}^\star$

$$\mathbf{p}^\star := \mathbf{p} - \alpha\,\mathbf{h}_\mathbf{p}^{-1}\,\mathbf{g}_\mathbf{p}, \tag{6}$$

where $\mathbf{g}_\mathbf{p} = \sum_{\tau\in\Omega_\mathbf{p}} \partial_\mathbf{p} E_\tau$ and $\mathbf{h}_\mathbf{p} = \sum_{\tau\in\Omega_\mathbf{p}} \mathbf{h}_{\tau,\mathbf{p}}$, a d-dimensional vector and $d\times d$ matrix respectively. We update the mesh one vertex at a time. The order is prioritized by the magnitude of the position change in the previous update.

The parameter α is the Newton step size. Independently at each vertex $\mathbf{p}$, we use a backtracking line search strategy to update it. We initialize $\alpha = 1$ and iteratively reduce it by 80% if the energy sum $E_\mathbf{p} = \sum_{\tau\in\Omega_\mathbf{p}} E_\tau$ increases or the volume of any simplex $\tau \in \Omega_\mathbf{p}$ becomes negative.

Boundary handling Vertices on a boundary surface should be restricted to remain on it. We therefore constrain the movement of a surface vertex $\mathbf{p}$ to its tangent plane on the reference surface. Let the basis vectors for an orthonormal system at $\mathbf{p}$ be denoted $\mathbf{U}_\mathbf{p}, \mathbf{V}_\mathbf{p}, \mathbf{N}_\mathbf{p}$ where $\mathbf{N}_\mathbf{p}$ is the unit surface normal. Inserting the proper constraints into Eq. 6, we obtain

$$\mathbf{p}^\star := \mathbf{p} - \alpha\,\begin{pmatrix}\mathbf{U}_\mathbf{p} & \mathbf{V}_\mathbf{p}\end{pmatrix}\,(\mathbf{h}_\mathbf{p}^S)^{-1}\,\mathbf{g}_\mathbf{p}^S,$$

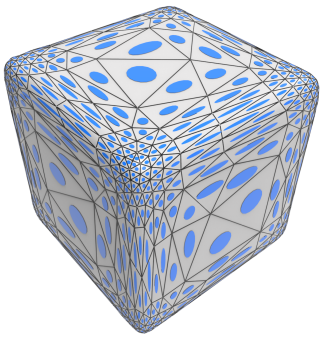


Figure 2: *Anisotropic meshing of a rounded cube. H_τ for each triangle is rendered as an ellipse.*

where $\mathbf{g}_\mathbf{p}^S = (\mathbf{U_p} \quad \mathbf{V_p})^T \mathbf{g_p}$ and $\mathbf{h}_\mathbf{p}^S = (\mathbf{U_p} \quad \mathbf{V_p})^T \mathbf{h_p}$ ($\mathbf{U_p} \quad \mathbf{V_p}$) are the surface tangent constrained versions of $\mathbf{g_p}$ and $\mathbf{h_p}$, a 2D vector and 2×2 matrix respectively. After moving $\mathbf{p}$ to $\mathbf{p}^\star$, we project $\mathbf{p}$ to its nearest point on the surface.

Vertices on boundary or sharp feature curves must be similarly restricted. If $\mathbf{U_p}$ is the tangent direction along the curve at $\mathbf{p}$, the update formula becomes

$$\mathbf{p}^\star := \mathbf{p} - \alpha\, \mathbf{U_p} \left(\mathbf{U_p}^T \mathbf{h_p}\, \mathbf{U_p}\right)^{-1} \left(\mathbf{U_p}^T \mathbf{g_p}\right)$$

We assume that feature and boundary curves are identified in the reference mesh. The end points of these lines are fixed during optimization; boundary vertices are only allowed to move along these curves.

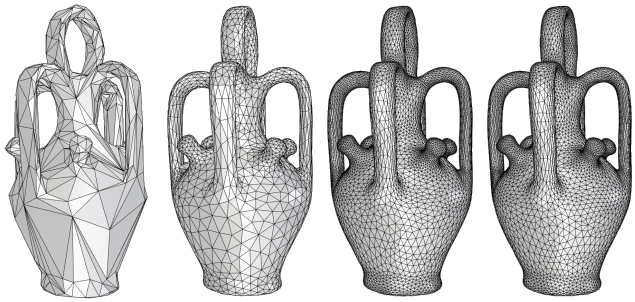
Connectivity update In adjacent triangles $\triangle_{\mathbf{p_0 p_1 p_2}}$ and $\triangle_{\mathbf{p_0 p_1 p_3}}$, we flip the edge $\overline{\mathbf{p_0 p_1}}$ if $E_{\triangle_{\mathbf{p_0 p_1 p_2}}} + E_{\triangle_{\mathbf{p_0 p_1 p_3}}} > E_{\triangle_{\mathbf{p_0 p_2 p_3}}} + E_{\triangle_{\mathbf{p_1 p_2 p_3}}}$. For tetrahedral meshes, many topological operators can be chosen to adapt mesh connectivity. We employ standard 2-3, 3-2, 4-4, and 2-2 flip operations [Klingner and Shewchuk 2007] to reduce energy. Edge flipping continues until no more edges can be flipped. The process must terminate because a positive energy is reduced each time and there are finitely many triangulations. To prevent degenerate simplices, we reject an edge flip if the volume of an adjacent simplex becomes smaller than 10^{-8} of the shape's bounding box volume or the edge is on a boundary or sharp feature curve.

Figure 2 shows an anisotropic meshing result of our LCT method. Triangle sizes and orientations follow the anisotropy, in this case derived from estimated curvature.

3.4 Alternating adjustment of other criteria

The previous section relaxes the mesh to match a specified anisotropy but ignores other criteria such as shape approximation error. We therefore add the following criteria and alternate in each iteration a pass to enforce them before a pass to reduce LCT energy. Our strategy is similar to [Tournois et al. 2009b], which alternates Delaunay refinement and ODT optimization. The algorithm terminates when all criteria are satisfied and LCT energy stops changing. Figure 3 shows an example.

Edge length regularization Under the inverse transformation induced by the augmented metric, we bound the target edge length L within $[L/\beta, \beta L]$. We split an edge if $|e_{\mathcal{M}^{-1}}| > \beta L$, and collapse it if $|e_{\mathcal{M}^{-1}}| < L/\beta$ and the collapse preserves mesh topology. We also reject the collapse if it brings the local geometric error above ϵ (see definition in the next paragraph). Enforcing $\beta = 1$ is too inflexible [Dobrzynski and Frey 2008]; we set $\beta = 1.5$. The order of edge split or collapse operations is governed by a priority queue of edges sorted by $|e_{\mathcal{M}^{-1}}|$. Riemannian edge length $|e_{\mathcal{M}^{-1}}|$ is computed



(a) **(b)** **(c)** **(d)**

Figure 3: *Progressive relaxation results combining steps of LCT energy minimization and refinement based on other geometric criteria in each iteration, on the Botijo model. (a) initial mesh (700 vertices, 1416 faces); (b) after first iteration (2730 vertices, 5476 faces); (c) after second iteration (10887 vertices, 21790 faces); (d) final result (11118 vertices, 22252 faces) after ten iterations.*

using the approximate formula given in Section 2, in terms of the inverse metric tensors at the edge's two endpoints.

Each regularization pass splits or collapses an edge only once. Subsequent LCT minimization further regularizes their lengths, so that later passes must handle progressively fewer irregular edges. This strategy saves computation especially in the common case where edge lengths are initially larger than targeted.

Geometric error control To control approximation error, we query the distance from the midpoint of every edge and the center of every face to the reference surface. If it exceeds ϵ, we split that edge or face by inserting its center point and projecting it back onto the reference. ϵ is set to $0.0001\, l$ where l is the diagonal of the domain's bounding box.

Sliver elimination Sliver tetrahedrons, which contain small dihedral angles after applying the inverse transformation ($\leq 15^\circ$ in our implementation), impair mesh quality. For example, they slow convergence of numerical PDE solvers computed over the mesh. Eliminating all slivers is hard both in theory and practice; various heuristics have been employed in literature. We combine the following operations to remove as many slivers as possible.

- Perform 5-4 flips, where the region formed by a sliver and its four neighboring tetrahedra contains at most 7 vertices; i.e, at least one neighboring vertex opposite to a sliver face is shared by multiple neighboring tetrahedra. In this case, we remove the sliver and re-tetrahedralize its neighborhood into 4 tetrahedra with a larger minimum dihedral angle.
- Perturb sliver vertices and perform flips to eliminate small dihedral angles. This is a straightforward application of [Tournois et al. 2009a] to anisotropic meshes.
- Remove boundary slivers (all of whose four vertices lie on the domain boundary) with nearly zero volume.
- Collapse one edge of a sliver if it increases the minimum dihedral angle across all neighbors.

Unlike the previous two mesh adjustments, this strategy is performed once as a post-process and successfully eliminates most slivers in our experiments.

4 Experiments and Comparisons

We compare our method with state-of-the-art algorithms, including a particle-based method (particle) [Zhong et al. 2013], discrete anisotropic centroidal Voronoi tessellation (ACVT) [Valette et al. 2008], anisotropic Delaunay refinement (ADR) [Boissonnat et al.

2014], bi-dimensional anisotropic mesh generation (BAMG) [Hecht 1998], and anisotropic tetrahedral remeshing/moving mesh generation (MMG3D) [Dobrzynski and Frey 2008]. The last two methods match anisotropy by edge split/collapse operations and Laplacian-like smoothing. We report timings and mesh quality statistics, defined below. Our method typically performs 10 or fewer passes alternating LCT optimization (Section 3.3) and mesh adjustment for other criteria (Section 3.4); LCT energy optimization in one pass typically converges in fewer than 100 iterations. The experiments were performed on a desktop PC with a 2.83GHz Intel Core Quad and 8GB of RAM.

Initial mesh For 2D meshing, the initial mesh is obtained via constrained triangulation of vertices on the specified polygonal domain boundary. For 3D surface meshing, a dense mesh is provided for estimating curvature, called the *reference mesh*. This reference is simplified to yield the initial mesh. The simplification level can be estimated by average anisotropic edge length. (The user can also directly specify the initial mesh, which should be a reasonable approximation to the reference to obtain good results.) All methods including ours utilize queries on the reference mesh, including evaluating interpolated metric tensors at arbitrary surface points and projecting arbitrary points onto the reference surface.

For 3D volumetric meshing, an input polyhedron specifies the domain boundary. We apply constrained Delaunay triangulation on it to obtain the initial mesh using TetGen software. The initial mesh includes only the boundary vertices and is later refined by our alternating algorithm.

In 2D and 3D surface comparisons, we invoke only LCT optimization without the other geometric criteria from Section 3.4. The desired number of vertices is determined by the available output of other methods, so the experimental task is remeshing requiring no refinement. We apply our method to the same initial mesh used by competing methods if this data is available to us. Otherwise, we randomly sample the desired number of points over the surface and generate a restricted Delaunay triangulation [Yan et al. 2009] on them as our initial mesh. In 3D volumetric meshing, we perform alternating refinement (as do other methods), since the initial mesh is very coarse and without interior vertices.

An input mesh much coarser or denser than the target requires many splits or collapses in edge regularization, leading to many relaxation iterations. A very coarse initial mesh also risks aliasing/undersampling of the desired anisotropy variation.

Riemannian metric The Riemannian metric is either specified as an analytic function or linearly interpolated from a reference mesh equipped with a tensor metric per vertex. We use $\delta = 10^{-8}$ in Eq. 1 to ensure $\mathbf{M}$ is positive-definite. For 3D surface meshing, we use the curvature tensor $\mathbf{M} = \mathbf{Q}\,\mathrm{diag}\,(\hat{\kappa}_1, \hat{\kappa}_2, 0)\,\mathbf{Q}^T$, where $\hat{\kappa}_i = \max(|\kappa_i|, 10^{-4})$ and κ_i are the principal curvatures. $\mathbf{Q} = [\mathbf{U}_1, \mathbf{U}_2, \mathbf{N}]$ where $\mathbf{U}_i$ are the principal curvature directions and $\mathbf{N}$ is the surface normal. The threshold 10^{-4} prevents the anisotropy from reaching 0 and generating zero-length edges. The curvature tensor is estimated from the vertices of the reference mesh using Rusinkiewicz's method [2004], and smoothed as suggested in [Boissonnat et al. 2014] to remove noise arising from the discrete estimation. Our implementation computes and smooths the curvature tensor in the same way regardless of whether the vertex is on a feature curve. (Note that our approach integrates curvature over each simplex anyway.) We obtain significantly increased vertex density around sharp features due to their large anisotropy (Figure 10). The target anisotropic edge length L is set to 1. Meshes at different levels of detail can be obtained simply by scaling L and adjusting the geometric error criterion ϵ.

Quality metrics Many anisotropic mesh quality metrics can be devised [Shewchuk 2002]; we use the ones from [Zhong et al. 2013]. For each triangle τ, we transform it back to the isotropic space, denoted τ^{-1}. We then measure: (1) its *angular quality* as the smallest angle of τ^{-1}, denoted $\theta(\tau)$; (2) its *triangle quality* $\xi(\tau) \triangleq 4\sqrt{3}\,a\,p/h$ where a is the area of τ^{-1}, p is its perimeter, and h its longest edge length; and (3) its *area quality* $\chi(\tau) \triangleq \frac{|\tau^{-1}|}{\sum_{\tau_k \in \mathcal{T}} |\tau_k^{-1}|/N}$. $\xi(\tau) = 1$ if τ^{-1} is a regular triangle. $\chi(\tau)$ evaluates the triangle's area uniformity in the mesh, with an optimum value of 1. We visualize area quality χ over the mesh, and report histograms and minimum (worst-case), average, and standard deviation over all mesh triangles for angular and triangle quality, denoted $\theta_{min}, \theta_{avg}, \theta_{dev}$, and $\xi_{min}, \xi_{avg}, \xi_{dev}$. We also report histograms over *all* simplex angles (not just the minimum), using the label θ^*. Finally, we report the ratio of valence-6 vertices, r_6, to measure mesh regularity, with an optimum value of 1.

For tetrahedral meshes, we similarly transform each tetrahedron to the isotropic space and measure its smallest dihedral angle $\theta(\tau)$, reporting its minimum and average values over the mesh. We also measure the *radius-edge ratio* $\rho(\tau) \triangleq r_{\tau^{-1}}/|e_{\tau^{-1}}|$ where $r_{\tau^{-1}}$ is the circumradius and $|e_{\tau^{-1}}|$ the shortest edge length of τ^{-1}. Their optimal values occur when τ^{-1} is a regular tetrahedron, yielding $\rho_{opt} = \frac{\sqrt{6}}{4} \approx 0.61$ and $\theta_{opt} = \cos^{-1}(\frac{1}{3}) \approx 70.53°$. Higher $\rho(\tau) > \rho_{opt}$ and lower $\theta(\tau) < \theta_{opt}$ values mean a worse (less regular) tetrahedron.

4.1 2D meshing

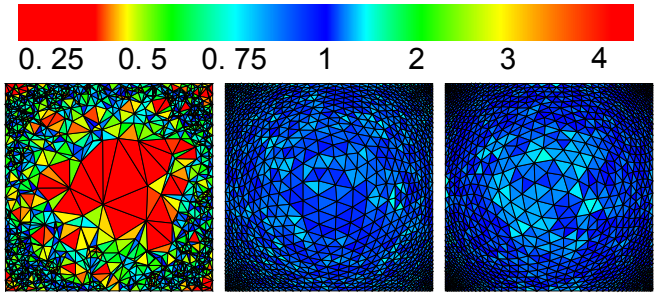
Example I Figure 4 shows an example whose Riemannian metric is induced by the function $u(x, y) = \exp((x^2 + y^2)/10)$. Since $u(x, y)$ is convex, we can apply both ODT and LCT minimization. Figure 4ac and Table 1 shows that their results are comparable, in terms of both ODT energy and our mesh quality measures.

We also compare Zhong et al.'s particle-based method [2013]. We use Zhong's recommended kernel width σ and choose two particle search sizes 5σ and 20σ for testing. The particle method's average triangle quality is worse than ODT or LCT, with particularly poor worst-case triangle quality (ξ_{min}). The false-color visualization in Figure 4de shows its poorer area quality. Note that particle energy penalizes non-uniformity in Riemannian distance along edges but ignores behavior within triangles.

Finally, we compare Chen's local patch method based on ODT minimization [2004]. It first constructs a convex quadratic function at a vertex $\mathbf{p}$ using the average metric $\overline{M}_\mathbf{p}$ in its one-ring neighborhood $\Omega_\mathbf{p}$, via $u(\mathbf{x}) = \mathbf{x}^T\,\overline{M}_\mathbf{p}\,\mathbf{x}$. It then sequentially updates each vertex $\mathbf{p}$ by optimizing the local ODT energy $\int_{\Omega_\mathbf{p}} |u(\mathbf{x}) - \hat{u}(\mathbf{x})|\,\mathrm{d}\mathbf{x}$. This method may appear similar to ours in its use of a local quadratic function. However, it does not consider the vertex update's effect on energy at other vertices. The algorithm may not converge or

Method	$E_{ODT,u,1}$	E_{max}	$\xi_{min}/\xi_{avg}/\xi_{dev}$	$\theta_{min}/\theta_{avg}/\theta_{dev}$
ODT	13.16	0.02	0.57/0.87/0.07	30.1°/49.3°/5.4°
LCT	**13.07**	**0.01**	**0.65/0.89/0.06**	**31.1°/50.7°/5.3°**
particle (5σ)	14.35	0.11	0.07/0.86/0.12	4.0°/49.7°/7.9°
particle (20σ)	14.79	0.20	0.04/0.85/0.12	2.1°/49.6°/7.8°
local patch	49.70	1.92	0.47/0.85/0.08	21.1°/47.4°/6.5°

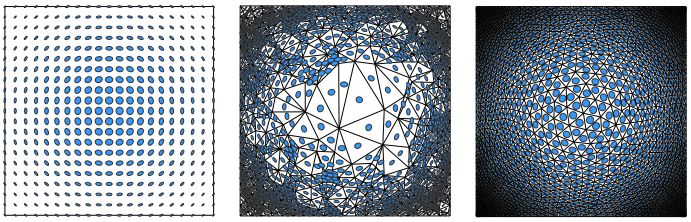
Table 1: *Example I – mesh quality comparisons with ODT, Zhong et al.'s particle method, and Chen's local patch method. E_{max} denotes the max ODT error over all triangles: $\max_{\tau \in \mathcal{T}} \int_\tau |\hat{u}(\mathbf{x}) - u(\mathbf{x})|\,\mathrm{d}\mathbf{x}$; lower energy is better. A number in boldface emphasizes the best result observed in the experiment.*



(a) *initial mesh* **(b)** *ODT* **(c)** *LCT*

(d) *particle* (5σ) **(e)** *particle* (20σ) **(f)** *local patch*

Figure 4: *Example I – meshing a 2D domain. We compare different methods on the convex function* $u(x, y) = e^{(x^2+y^2)/10}$ *over the domain* $[-5.5, 5.5]^2$. *Area quality* χ *is color-coded based on the color map shown at top. Dark blue is optimal.*



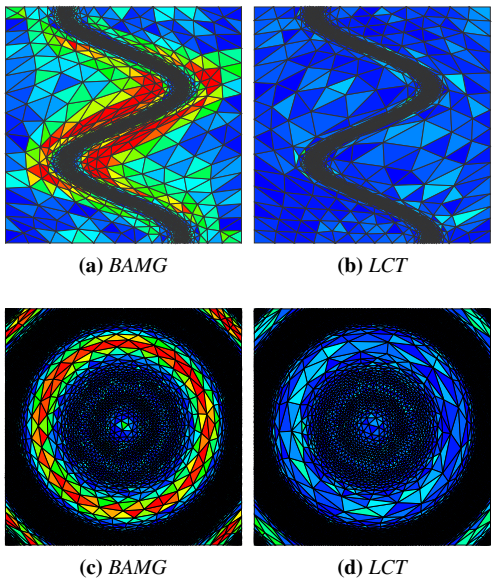
(a) *target* **(b)** *initial* **(c)** *final*

Figure 5: *Anisotropy visualization in example I. (a) Targeted anisotropy on the square domain, rendered as ellipses. (b)* H_τ *on the initial mesh. (c)* H_τ *after convergence. The resulting mesh yields anisotropy close to the target.*

it converges to a non-optimal anisotropic match. Figure 4f shows that Chen's method does not fit the anisotropy well in terms of area quality, and yields much higher ODT energy than other methods. We note that Chens method can be improved by optimizing the sum over all local ODT energies affected by a vertex perturbation; i.e. energies defined over its entire one-ring neighborhood. This essentially yields a variant of our LCT method in which the quadratic functions are determined per-vertex rather than per-simplex. Figure 5 visualizes the anisotropy of this example on the initial mesh and our final result.

Example II Figure 6 compares our method with BAMG, a popular program for 2D anisotropic meshing. We choose a square as the domain and test two different Riemannian metrics from non-convex functions. BAMG produces low-quality triangles especially in regions where the anisotropy changes rapidly. We take BAMG's output as our initial mesh and apply LCT optimization to obtain clear improvement in mesh quality (see also Table 2).

Example III Figure 7 shows another comparison with Zhong's particle method. On a square domain $[-100, 100]^2$, we specify a Riemannian metric via the circular anisotropic tensor field: $\mathbf{M}(\mathbf{x}) = Q(\mathbf{x})\ \mathrm{diag}(\lambda^2(\|\mathbf{x}\|), 1)\, Q^T(\mathbf{x})$ where $\lambda \in [1, 10]$. 20000 points are sampled within the domain. Our method takes 31 seconds to converge while Zhong's takes about 20 minutes as reported in [2013].



(a) *BAMG* **(b)** *LCT*

(c) *BAMG* **(d)** *LCT*

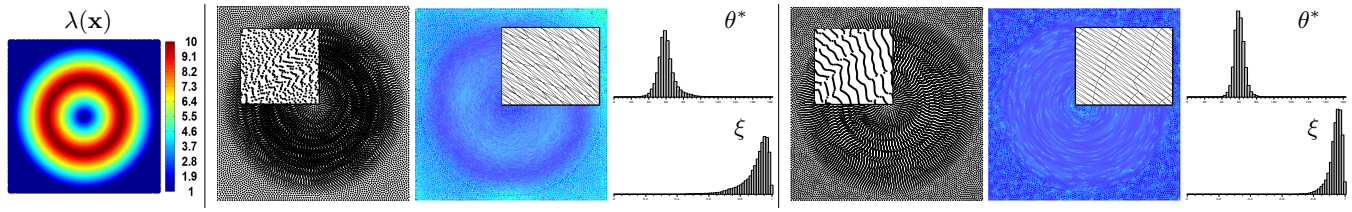
Figure 6: *Example II – comparison with BAMG. The Riemannian metric is determined from the non-convex analytic functions* $u(x, y) = \tanh(10(\sin(5y) - 2x)) + x^2y + y^3$ *(upper row) and* $u(x, y) = e^{3\cos\frac{x^2+y^2}{5}}$ *(lower row) over the domain* $[-5.5, 5.5]^2$, *with anisotropy ratios in* $[1.9, 394.4]$ *and* $[5.4, 597.8]$, *respectively. Mesh triangles are colored by area quality,* χ.

It is visually clear that our result achieves a better point distribution and better mesh quality (see also Table 2).

Example IV LCT energy minimization is a kind of mesh smoothing, so it is natural to compare it with other smoothing functions. We chose three popular ones. The first sums the squared anisotropic length over all edges: $E_1 \triangleq \sum |\mathbf{e}_M|^2$. The second sums squared anisotropic lengths of all simplex edges normalized by the simplex's anisotropic area: $E_2 \triangleq \sum_\tau \frac{\sum_{\mathbf{e}\in\tau} |\mathbf{e}_M|^2}{|\tau_M|}$, yielding a measure of the anisotropic transformation's distortion that relates to the maximum eigenvalue of the element stiffness matrix of τ_M [Shewchuk 2002; Clark et al. 2012]. The third sums the product of anisotropic lengths of all simplex edges normalized by the simplex's anisotropic area: $E_3 \triangleq \sum_\tau \frac{\prod_{\mathbf{e}\in\tau} |\mathbf{e}_M|}{|\tau_M|}$ [Shewchuk 2002]. We substitute E_1, E_2 or E_3 for LCT energy in the vertex update and selection of edge flips from Section 3.3, starting from the same initial mesh generated by our edge length regularization strategy. Figure 8 and Table 2 show that optimizing E_1 or E_2 improves mesh quality much less than optimizing LCT energy, while optimizing E_3 actually degrades quality of the initial mesh.

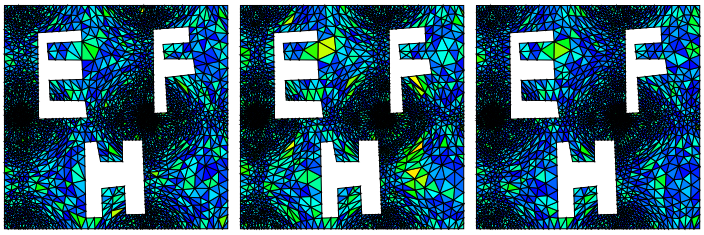
	#vert	λ	$\xi_{min}/\xi_{avg}/\xi_{dev}$	$\theta_{min}/\theta_{avg}/\theta_{dev}$	r_6
Fig. 6a (BAMG)	1289	[1.9,394.4]	0.22/0.83/0.13	10.3°/46.4°/8.9°	0.60
Fig. 6b (LCT)	1289	[1.9,394.4]	**0.42/0.89/0.08**	**22.8°/50.4°/5.8°**	**0.69**
Fig. 6c (BAMG)	6251	[5.4,597.8]	0.07/0.87/0.08	3.9°/49.6°/6.0°	0.60
Fig. 6d (LCT)	6251	[5.4,597.8]	**0.45/0.90/0.07**	**21.1°/51.3°/5.2°**	**0.70**
Fig. 7 (particle)	20000	[1,10]	0.09/0.90/0.08	6.1°/52.5°/5.0°	0.78
Fig. 7 (LCT)	20000	[1,10]	**0.57/0.94/0.04**	**31.0°/54.5°/3.3°**	**0.90**
Fig. 8a (init)	2316	[1,429]	0.32/0.80/0.11	14.5°/44.4°/7.3°	**0.67**
Fig. 8b (E_1)	2316	[1,429]	0.38/0.86/0.09	23.4°/48.5°/6.3°	0.40
Fig. 8c (E_2)	2316	[1,429]	0.42/0.87/0.08	25.4°/49.4°/5.6°	0.41
Fig. 8d (E_3)	2316	[1,429]	0.15/0.80/0.13	6.4°/44.4°/8.7°	0.38
Fig. 8e (LCT)	2316	[1,429]	**0.60/0.91/0.06**	**32.5°/52.6°/4.6°**	0.64

Table 2: *Quality metrics for examples II, III and IV.*

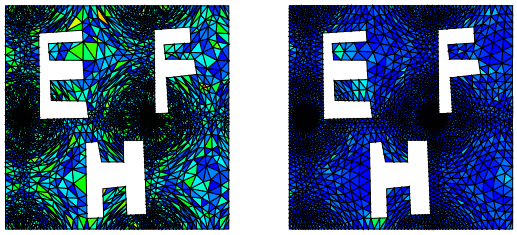


(a) *anisotropy* **(b)** *particle* **(c)** *LCT*

Figure 7: *Example III – comparison between Zhong et al. 's particle method (b) and our method (c) using a metric with circularly symmetric anisotropy $\lambda(\mathbf{x})$ shown in (a). From left to right, we compare point distributions, color-coded area quality χ, and histograms of angle and triangle quality. Zoomed-in insets of the point distribution and mesh show that our result transitions more smoothly than the particle-based method to adapt to changes in anisotropy, and generates a more regular mesh. It also yields better distributions of angle θ^* (clustered more tightly around 60°), and triangle quality ξ (clustered more tightly around 1).*



(a) *initial mesh* **(b)** *optimizing E_1* **(c)** *optimizing E_2*



(d) *optimizing E_3* **(e)** *optimizing LCT*

Figure 8: *Example IV – comparison to other smoothing functions. The Riemannian metric is derived from $u(x, y) = e^{\sin x + \cos y}$ with anisotropy ratio in $[1, 429]$. Meshes are colored by area quality, χ.*

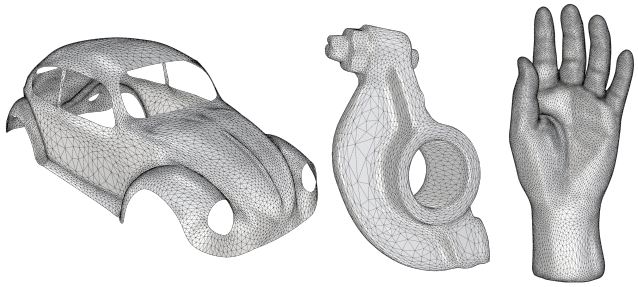


Figure 9: *Anisotropic 3D surface meshes (Beetle, Rockarm and Hand) generated by our method.*

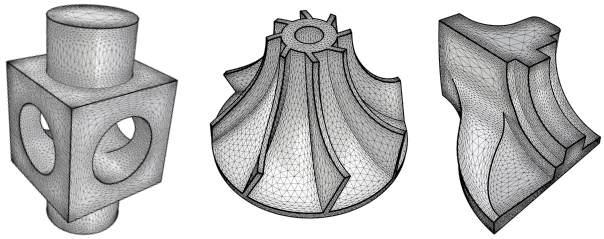


Figure 10: *Anisotropic 3D surface meshes with sharp features (Block, Impeller and Fandisk) generated by our method.*

4.2 3D surface meshing

Figure 9 shows some results (Beetle, Rockarm, Hand models) with anisotropy specified by the surface's curvature tensor. Our method takes 64, 17 and 77 seconds respectively for these models and achieves high quality as shown in Table 3. We also apply our method to surfaces with sharp features in Figure 10.

Figure 11 compares anisotropic centroidal Voronoi tessellation (ACVT) [Valette et al. 2008], a particle-based method [Zhong et al. 2013], and our method, on the Cyclide model. ACVT produces a poor result: 16.8% of triangles have angles less than 30°. The particle result is better, but still falls short. For instance, our ratio of valence-6 vertices represents a significant improvement over the particle method and ACVT: 0.87 versus 0.78 and 0.51. We obtain better triangle and angular quality too; see Table 3 for further data.

Figure 12 compares our method with ACVT, the particle-based method, and anisotropic Delaunay refinement (ADR) [Boissonnat et al. 2014] on the Fertility model. All methods target anisotropy from surface curvature. The particle-based method applies a 6D metric in terms of vertex normals and positions as recommended for it in [Zhong et al. 2013] while the other methods use standard 3×3 curvature tensors. The particle-based method runs 100 iterations and is accelerated by parallel computation. The targeted number of output mesh vertices is set to ADR's output. Our mesh yields better mesh regularity, and improved faithfulness to the original shape (see also Hausdorff errors in Table 3). It also better matches the curvature-based anisotropy, yielding improved angular and triangle quality.

Figures 1-middle and 13 show two results for more complex meshes.

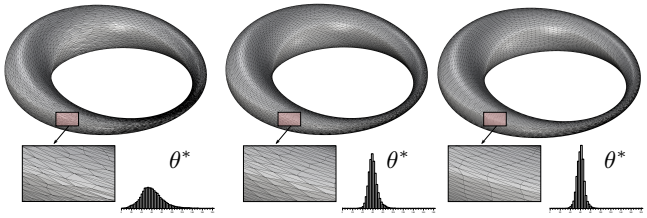
4.3 Tetrahedral meshing

Figures 14 and 15 compare our method with MMG3D. The domain is the cube $[0.1, 1.1]^3$ (Figure 14) or $[1, 11]^3$ (Figure 15). Mesh quality of the two results is comparable; see Table 4. We achieve more optimal angular and radius-edge quality (ρ) than MMG3D, as can be seen in the histogram and table statistics. Our result's standard deviations of quality metrics are also smaller than MMG3D's. Note that our tetrahedral meshes are significantly sparser than MMG3D's (1870 vertices versus 2365 vertices in Figure 14, 6338 vertices versus 8217 vertices in Figure 15). Figure 16 shows another meshing comparison with MMG3D on a unit-sphere domain. The same surface mesh was used as input to both methods. Our result has fewer silvers and better angular and radius-edge quality.

Our algorithm is slower than MMG3D because of our inefficient

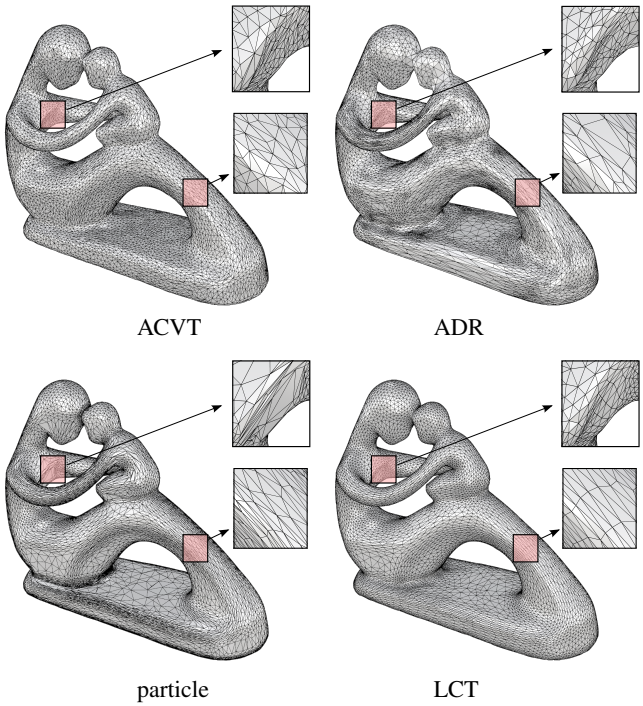
Model	ref #vert	init #vert	#vert	λ	$\xi_{min}/\xi_{avg}/\xi_{dev}$	$\theta_{min}/\theta_{avg}/\theta_{dev}$	$\%_{<30^\circ}$	D_H	r_6	time (s)
Cyclide (particle)	-	8000	8000	[2, 29]	0.09/0.87/-	$5.03^\circ/49.7^\circ/-^\circ$	**0.04%**	4.4e-4	0.78	155.8
Cyclide (ACVT)	414720	8000	8009	[2, 29]	0.002/0.75/0.17	$0.07^\circ/40.8^\circ/10.8^\circ$	16.8%	3.6e-3	0.51	177.0
Cyclide (LCT)	25920	8000	8000	[2, 29]	**0.60/0.92/0.05**	$\mathbf{26.9^\circ/53.1^\circ/4.2^\circ}$	**0.03%**	**3.4e-4**	**0.87**	**17.5**
Fertility (ACVT)	223626	12480	12480	[1, 14]	0.00/0.67/0.19	$0.12^\circ/35.6^\circ/11.7^\circ$	32.67%	**1.1e-3**	0.39	37.5
Fertility (ADR)	-	12480	12480	[1, 14]	0.002/0.56/-	$0.06^\circ/29.9^\circ/-^\circ$	41.79%	5.8e-3	0.46	-
Fertility (particle)	-	12480	12301	[1, 14]	0.02/0.70/-	$0.86^\circ/37.6^\circ/-^\circ$	26.99%	2.3e-3	0.49	**10.0**
Fertility (LCT)	13971	12480	12480	[1, 14]	**0.54/0.89/0.07**	$\mathbf{24.9^\circ/50.8^\circ/5.1^\circ}$	**0.04%**	**1.1e-3**	**0.68**	66.8
Rockarm	9413	1272	5550	[1, 18]	0.34/0.86/0.08	$20.4^\circ/48.9^\circ/6.1^\circ$	0.46%	1.8e-3	0.60	17.6
Fandisk	6475	1927	7950	[1, 15]	0.14/0.87/0.08	$8.4^\circ/48.9^\circ/5.8^\circ$	0.18%	9.5e-4	0.60	19.6
Beetle	17908	17908	9817	[1, 15]	0.27/0.87/0.08	$13.2^\circ/48.9^\circ/6.0^\circ$	0.66%	9.2e-4	0.52	64.3
Block	8052	3307	11667	[1, 15]	0.51/0.88/0.07	$27.2^\circ/50.1^\circ/5.4^\circ$	0.03%	1.1e-3	0.63	38.7
Impeller	10000	10000	11737	[1, 16]	0.39/0.87/0.08	$22.1^\circ/49.6^\circ/5.8^\circ$	0.17%	6.5e-4	0.60	110.5
Botijo	14989	700	13890	[1, 16]	0.52/0.89/0.07	$23.1^\circ/50.9^\circ/5.0^\circ$	0.04%	1.6e-3	0.66	39.1
Hand	30000	2576	21226	[1, 14]	0.46/0.90/0.06	$20.6^\circ/51.4^\circ/4.8^\circ$	0.04%	1.1e-3	0.67	77.1
Buddha	115474	5000	63284	[1, 34]	0.41/0.88/0.07	$17.4^\circ/49.9^\circ/5.4^\circ$	0.03%	6.7e-4	0.64	178.5
Lucy	262787	74119	255097	[1, 19]	0.26/0.90/0.06	$15.4^\circ/51.4^\circ/5.0^\circ$	0.04%	3.9e-4	0.70	2766.0

Table 3: *Statistics and timings for surface meshing. We report the number of vertices in the reference mesh ("ref #vert"), initial mesh ("init #vert"), and output mesh ("#vert"). λ reports the range of anisotropy ratios. $\%_{<30^\circ}$ is the fraction of triangles whose minimal angle is smaller than 30°. D_H reports the maximum Hausdorff distance between the reference and output meshes with respect to the diagonal of the reference's bounding box. Reference meshes for the ACVT method were subdivided to provide better approximation accuracy. Data for the particle and ADR methods is copied from [Zhong et al. 2013].*



(a) ACVT **(b)** particle **(c)** LCT

Figure 11: *Comparison with ACVT and particle methods on the Cyclide model.*



ACVT ADR

particle LCT

Figure 12: *Comparison with ACVT [Valette et al. 2008], particle-based method [Zhong et al. 2013], and ADR [Boissonnat et al. 2014] on the Fertility model.*

edge flip implementation and sequential vertex update.

We also applied 100 iterations of LCT optimization and a final sliver elimination pass to MMG3D's result. Quality metrics for this result are listed in Table 4 in rows labeled "MMG3D-LCT". Further LCT iterations improve both angular and radius-edge quality.

Figure 1-right and 17 show two more volumetric meshing results. The 3D domains are a cube and a bumpy shape respectively, with anisotropy specified via analytic functions. The targeted anisotropy in Figure 1-right is $\mathbf{M}(\mathbf{x}) = \mathbf{Q}^T(\mathbf{x})\,\Lambda^2(\mathbf{x})\,\mathbf{Q}(\mathbf{x})$, where $\Lambda(\mathbf{x}) = \text{diag}\left((0.025 + (1 - e^{-0.01|\|\mathbf{x}\|^2 - 49|}))^{-1}, 1, 1\right)$ and $\mathbf{Q}$'s three columns are $\mathbf{x}/\|\mathbf{x}\|$ and two orthogonal vectors. The domain is the cube $[1, 11]^3$.

In communication with the authors, we also attempted to compare anisotropic Delaunay refinement [Boissonnat et al. 2011] on the example in Figure 1-right. ADR provides a theoretical guarantee on certain mesh qualities. However, their implementation is still under development and was unable to generate a result.

5 Conclusion

Locally convex optimal triangulation provides a novel and simple way to generate high-quality anisotropic simplicial meshes in 2D/3D surface or 3D volumetric domains. Our method inherits the advantages of optimal Delaunay triangulation but extends that technique to general Riemannian metrics. It provides good performance and excellent overall mesh quality superior to previous methods. We also note a few limitations to address in future work.

Quality bound Compared to [Labelle and Shewchuk 2003; Boissonnat et al. 2008a], we do not provide a theoretical guarantee on mesh quality. It would be interesting to try alternate steps of [Boissonnat et al. 2008a]'s anisotropic Delaunay refinement and our LCT optimization to bound worst-case mesh quality while preserving our method's average-case quality. Minimizing the maximum error over all simplices, $E_{LCT,\infty}$, is another potential way to control worst-case quality.

Geometric and anisotropic incompatibility When the specified anisotropy is derived from curvature tensors on the underlying 3D surface domain, geometric error control and LCT energy minimization are compatible. In other words, meshing guided by curvature yields a good geometric approximation. But one can also specify an anisotropy not related to the domain's curvature. In this case, regions

Figure 13: *Anisotropic meshing of the Lucy model.*

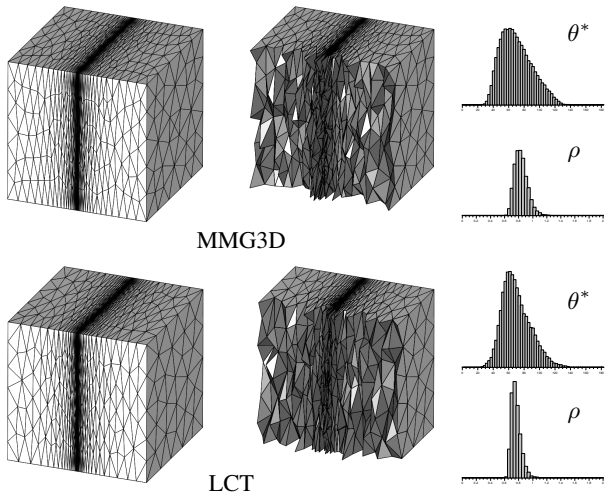


Figure 14: *Simple volumetric example with anisotropy variation in a single direction. The Riemannian metric is* $\mathbf{M}(\mathbf{x}) = \Lambda^2(\mathbf{x})$*, where* $\Lambda(\mathbf{x}) = \mathrm{diag}\left((0.0025 + 0.2(1 - e^{-|x-0.6|}))^{-1}, 5, 5\right)$*. Middle images are sections through the tetrahedral meshes. Right images show histograms of all dihedral angles* θ^* *and radius-edge quality* ρ*. Our method produces tighter distributions of angles and radius-edge ratios around their optimal values (70.5° for* θ^**, 0.61 for* ρ*).*

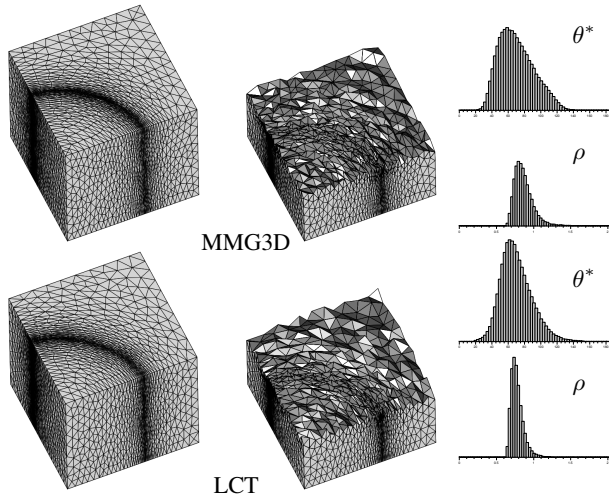


Figure 15: *Another simple volumetric example with cylindrical variation of anisotropy. The Riemannian metric is* $\mathbf{M}(\mathbf{x}) = \mathbf{Q}^T(\mathbf{x})\,\Lambda^2(\mathbf{x})\,\mathbf{Q}(\mathbf{x})$*, where* $\Lambda(\mathbf{x}) = \mathrm{diag}\left(2(0.1 + 2(1 - e^{-0.01|x^2+y^2-49|}))^{-1}, 1, 1\right)$ *and* $\mathbf{Q}$*'s three columns are* $(x/\sqrt{x^2+y^2}, y/\sqrt{x^2+y^2}, 0)^T$*,* $(-y/\sqrt{x^2+y^2}, x/\sqrt{x^2+y^2}, 0)^T$ *and* $(0, 0, 1)^T$*.*

Model	init #vert	#vert	#tet	λ	$\theta_{min}/\theta_{avg}/\theta_{dev}$	$\rho_{max}/\rho_{avg}/\rho_{dev}$	#sliver$_b$	#sliver$_a$	time (s)
Fig. 14 (LCT)	948	**1870**	**8144**	$[1, 80]$	**24.7°**/**51.7°**/7.5°	1.46/**0.76**/0.07	71	0	8.3
Fig. 14 (MMG3D)	-	2365	10860	$[1, 80]$	22.0°/48.1°/7.7°	1.37/0.83/0.09	-	0	**4.9**
MMG3D-LCT	-	2365	10913	$[1, 80]$	22.6°/51.5°/7.8°	**1.36**/0.77/0.07	23	0	5.1
Fig. 15 (LCT)	2226	**6338**	**31840**	$[1, 20]$	17.7°/50.6°/8.5°	**1.54**/0.78/0.08	517	0	72.6
Fig. 15 (MMG3D)	-	8217	42067	$[1, 20]$	16.2°/46.8°/8.2°	2.59/0.86/0.13	-	0	**15.4**
MMG3D-LCT	-	8217	42435	$[1, 20]$	**18.1°**/**50.9°**/8.6°	1.71/**0.77**/0.09	530	0	31.2
Fig. 16 (LCT)	1966	**4739**	**22427**	$[1, 10]$	9.1°/44.7°/10.7°	5.56/0.89/0.18	1715	72	103.8
Fig. 16 (MMG3D)	-	5187	25311	$[1, 10]$	6.2°/37.1°/10.9°	6.27/1.21/0.41	-	264	**6.5**
MMG3D-LCT	-	5187	25767	$[1, 10]$	**9.4°**/**44.8°**/10.8°	**3.50**/**0.88**/0.17	1932	**65**	33.1
Fig. 1-right	2158	6554	32668	$[1, 40]$	15.3°/48.9°/9.2°	2.80/0.83/0.13	844	0	104.6
Fig. 17	18183	35096	153959	$[1, 13]$	15.3°/51.1°/8.3°	1.41/0.77/0.07	534	0	339.4

Table 4: *Statistics and timings for tetrahedral meshing. #sliver$_b$ and #sliver$_a$ report the number of slivers before and after applying our sliver elimination strategy.*

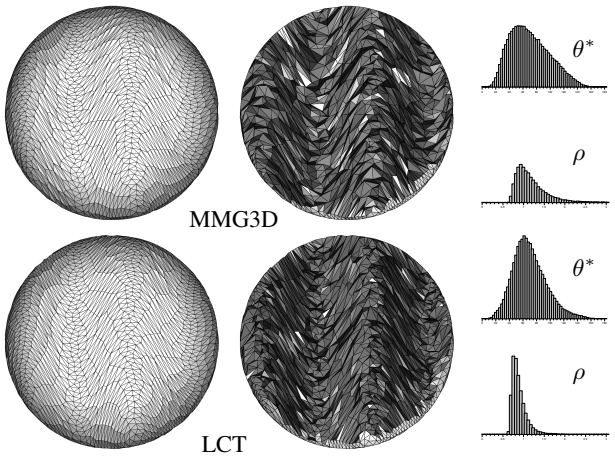


Figure 16: *Sinusoidal anisotropy variation in a ball. The Riemannian metric is* $\mathbf{M}(\mathbf{x}) = \mathbf{Q}^T(\mathbf{x})\,\Lambda\,\mathbf{Q}(\mathbf{x})$*, where* $\Lambda = \mathrm{diag}(1000, 10, 10)$ *and* $\mathbf{Q}$*'s three columns are* $(2\cos(6x), 1, 0)^T$ *and two orthogonal vectors.*

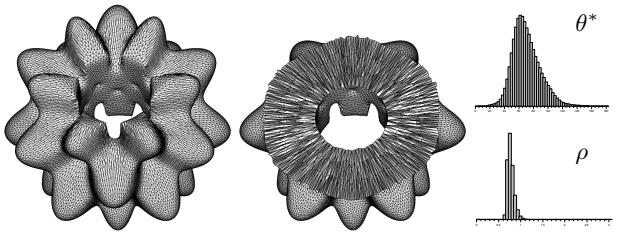


Figure 17: *Cylindrical anisotropy variation in a complex bumpy domain. The targeted anisotropy uses basis vectors* $\mathbf{Q}$ *as in Figure 15, but with* $\Lambda_1(\mathbf{x}) = 1.2/(0.5 + 1 - e^{-0.05(x^2+y^2-2.56)})$, $\Lambda_2(\mathbf{x}) = \Lambda_3(\mathbf{x}) = \Lambda_1(\mathbf{x})\,(1 + 5\sqrt{x^2+y^2})$.

of high geometric error keep getting refined while LCT minimization encourages the newly-created vertices to flow toward (different) regions of high anisotropy. Vertex density can grow unnecessarily over the whole mesh. Fortunately, this oversampling is prevented by our loose edge regularization which avoids splitting an edge unless its inversely transformed length gets bigger than βL. It may also be helpful to apply Tournois et al.'s locking strategy which deactivates vertices whose neighborhood has already attained sufficient mesh quality and geometric fidelity, updating only the remainder [2009b]. A more consistent solution would be to balance both metrics in the objective by adding LCT energy for the anisotropy match to an energy accounting for geometric error.

Convex local functions As with Chen et al.'s approach [Chen et al. 2007], our method converts a negative-definite Hessian to a positive-definite one via Eq. 1 (note the absolute value on Λ). This decreases fidelity when the u function locally determined by the specified metric is nonconvex. It is possible to use a general rather than positive-definite H in Eq. 3. Such a generalized LCT formulation might provide increased fidelity but also complicates optimization, since the absolute value operator in Eq. 4 no longer vanishes and Eq. 5 is no longer valid. Another future direction is to replace quadratic with more general convex functions that better fit the local tensor field, e.g. a simplicial Bernstein-Bézier spline.

Semi-regular meshes Panozzo et al. [2014] recently proposed an anisotropic quadrilateral meshing method that warps a frame field to a cross field (having orthogonal tangent vectors), computes an isotropic quadrilateral mesh on this deformed mesh, and transforms the result back to the original space. This approach could potentially be extended to generate anisotropic, semi-regular triangle meshes using a 6-Rosy frame field. Li et al. [2014] propose another anisotropic meshing method that computes a tensor-guided quadrilateral mesh and splits quads to triangles. Both methods input a smooth frame field containing only a few singular points. They are unable to handle large anisotropy variation, can produce some poor-quality mesh elements, and can fail in the parametrization step due to "flipovers". Our method does not explicitly limit the number of irregular vertices but still produces fewer than other methods not based on semi-regular meshing (see higher r_6 in Table 3). An interesting extension would be to detect and eliminate irregular vertices during its edge flipping, splitting and merging steps.

Acknowledgements

Original models are courtesy of the Aim@Shape Repository and the Stanford 3D Scanning Repository. Particle-based results in Figures 7, 11, 12 were provided by the authors of [Zhong et al. 2013]. We used the implementation of [Valette et al. 2008] to compute ACVT's results. BAMG and MMG3D results were computed via FreeFem++ (`http://www.freefem.org`) and Gmsh (`http://geuz.org/gmsh`), respectively. We used the OpenMesh library (`www.openmesh.org`) for triangle meshing and adapted OpenVolumeMesh (`http://www.openvolumemesh.org`) for tetrahedral meshing. CGAL's AABB tree (`http://www.cgal.org`) was used to efficiently project vertices onto the reference surface and the TetGen library (`http://www.tetgen.org`) to compute initial tetrahedral meshes. Thanks to Mariette Yvinec and Mael Rouxel-Labbe for testing their ADR implementation.

References

ALLIEZ, P., COHEN-STEINER, D., YVINEC, M., AND DESBRUN, M. 2005. Variational tetrahedral meshing. *ACM Trans. Graph. (SIGGRAPH) 24*, 3, 617–625.

AMARI, S.-I., AND ARMSTRONG, J. 2014. Curvature of Hessian mnifolds. *Differential Geom. Appl.*, 33, 1–12.

BOISSONNAT, J.-D., COHEN-STEINER, D., AND YVINEC, M. 2008. Comparison of algorithms for anisotropic meshing and adaptive refinement. Tech. rep., INRIA. ACS-TR-362603.

BOISSONNAT, J.-D., WORMSER, C., AND YVINEC, M. 2008. Locally uniform anisotropic meshing. In *SOCG*, 270–277.

BOISSONNAT, J.-D., WORMSER, C., AND YVINEC, M. 2011. Anisotropic Delaunay mesh generation. Tech. rep., INRIA. INRIA-00615486.

BOISSONNAT, J.-D., SHI, K.-L., TOURNOIS, J., AND YVINEC, M. 2014. Anisotropic Delaunay meshes of surfaces. *ACM Trans. Graph.*, to appear.

CANAS, G. D., AND GORTLER, S. J. 2011. Orphan-free anisotropic Voronoi diagrams. *Discrete Comput. Geom. 46*, 3, 526–541.

CHEN, L., AND HOLST, M. 2011. Efficient mesh optimization schemes based on optimal Delaunay triangulations. *Comput. Methods in Appl. Mech. Eng. 200*, 912, 967–984.

CHEN, L., AND XU, J. 2004. Optimal Delaunay triangulations. *J. Comput. Math. 22*, 299–308.

CHEN, L., SUN, P., AND XU, J. 2007. Optimal anisotropic meshes for minimizing interpolation errors in L^p-norm. *Math. Comp. 76*, 179–204.

CHEN, L. 2004. Mesh smoothing schemes based on optimal Delaunay triangulations. In *Int. Meshing Roundtable*, 109–120.

CHENG, S.-W., DEY, T. K., RAMOS, E. A., AND WENGER, R. 2006. Anisotropic surface meshing. In *SODA*, 202–211.

CLARK, B., RAY, N., AND JIAO, X. 2012. Surface mesh optimization, adaption, and untangling with high-order accuracy. In *Int. Meshing Roundtable*. 385–402.

DESBRUN, M., DONALDSON, R. D., AND OWHADI, H. 2013. Modeling across scales: discrete geometric structures in homogenization and inverse homogenization. In *Multiscale Analysis and Nonlinear Dynamics*, 19–64.

DOBRZYNSKI, C., AND FREY, P. 2008. Anisotropic Delaunay mesh adaptation for unsteady simulations. In *Int. Meshing Roundtable*, 177–194.

DU, Q., AND WANG, D. 2005. Anisotropic centroidal Voronoi tessellations and their applications. *SIAM J. Sci. Comput. 26*, 3, 737–761.

FREY, P., AND GEORGE, P.-L. 2008. *Mesh Generation : Application to finite elements*, 2 ed. Wiley-ISTE.

GENZ, A., AND COOLS, R. 2003. An adaptive numerical cubature algorithm for simplices. *ACM Trans. Math. Softw. 29*, 3, 297–308.

GOES, F. D., MEMARI, P., MULLEN, P., AND DESBRUN, M. 2014. Weighted triangulations for geometry processing. *ACM Trans. Graph. 33*, 3, 28:1–28:13.

HECHT, F., 1998. BAMG: Bidimensional anisotropic mesh generator. http://www.ann.jussieu.fr/ hecht/ftp/bamg.

JIAO, X., COLOMBI, A., NI, X., AND HART, J. 2010. Anisotropic mesh adaptation for evolving triangulated surfaces. *Eng. with Comput. 26*, 4, 363–376.

KLINGNER, B. M., AND SHEWCHUK, J. R. 2007. Agressive tetrahedral mesh improvement. In *Int. Meshing Roundtable*, 3–23.

LABELLE, F., AND SHEWCHUK, J. R. 2003. Anisotropic Voronoi diagrams and guaranteed-quality anisotropic mesh generation. In *SOCG*, 191–200.

LÉVY, B., AND BONNEEL, N. 2012. Variational anisotropic surface meshing with Voronoi parallel linear enumeration. In *Int. Meshing Roundtable*, 349–366.

LÉVY, B., AND LIU, Y. 2010. L_p centroidal Voronoi tessellation and its applications. *ACM Trans. Graph. (SIGGRAPH) 29*, 4, 119:1–119:11.

LI, Y., LIU, Y., AND WANG, W. 2014. Planar hexagonal meshing for architecture. *IEEE. T. Vis. Comput. Gr.*, to appear.

LIU, Y., PAN, H., SNYDER, J., WANG, W., AND GUO, B. 2013. Computing self-supporting surfaces by regular triangulation. *ACM Trans. Graph. (SIGGRAPH) 32*, 4, 92:1–92:10.

MULLEN, P., MEMARI, P., DE GOES, F., AND DESBRUN, M. 2011. HOT: Hodge-optimized triangulations. *ACM Trans. Graph. (SIGGRAPH) 30*, 4, 103:1–103:12.

PANOZZO, D., PUPPO, E., TARINI, M., AND SORKINE-HORNUNG, O. 2014. Frame fields: anisotropic and non-orthogonal cross fields. *ACM Trans. Graph. (SIGGRAPH) 33*, 4, 134:1–134:11.

PERSSON, P.-O., AND STRANG, G. 2004. A simple mesh generator in MATLAB. *SIAM Rev. 46*, 329–345.

RUSINKIEWICZ, S. 2004. Estimating curvatures and their derivatives on triangle meshes. In *3DPVT*, 486–493.

SHEWCHUK, J. R., 2002. What is a good linear finite element? Interpolation, conditioning, anisotropy, and quality measures.

SHIMADA, K., YAMADA, A., AND ITOH, T. 2000. Anisotropic triangulation of parametric surfaces via close packing of ellipsoids. *Int. J. Comput. Geom. Ap. 10*, 4, 417–440.

THOMPSON, J. F., SONI, B. K., AND WEATHERILL, N. P., Eds. 1998. *Handbook of Grid Generation*. Wiley-ISTE.

TOURNOIS, J., SRINIVASAN, R., AND ALLIEZ, P. 2009. Perturbing slivers in 3D Delaunay meshes. In *Int. Meshing Roundtable*, 157–173.

TOURNOIS, J., WORMSER, C., ALLIEZ, P., AND DESBRUN, M. 2009. Interleaving Delaunay refinement and optimization for practical isotropic tetrahedron mesh generation. *ACM Trans. Graph. (SIGGRAPH) 28*, 3, 75:1–75:9.

VALETTE, S., CHASSERY, J.-M., AND PROST, R. 2008. Generic remeshing of 3D triangular meshes with metric-dependent discrete Voronoi diagrams. *IEEE. T. Vis. Comput. Gr. 14*, 2, 369–381.

YAN, D.-M., LÉVY, B., LIU, Y., SUN, F., AND WANG, W. 2009. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. *Comput. Graph. FORUM 28*, 5, 1445–1454.

ZHONG, Z., GUO, X., WANG, W., LÉVY, B., SUN, F., LIU, Y., AND MAO, W. 2013. Particle-based anisotropic surface meshing. *ACM Trans. Graph. (SIGGRAPH) 32*, 4, 99:1–99:14.

ZIENKIEWICZ, O. C., TAYLOR, R. L., AND ZHU, J. 2005. *The Finite Element Method: Its Basis and Fundamentals*, 6 ed. Butterworth-Heinemann.